A decentralized and robust approach to estimating a probabilistic mixture model for structuring distributed data

Ali El Attar, Antoine Pigeau and Marc Gelgon
LINA (UMR CNRS 6241)
Université de Nantes
Nantes, France
firstname.name@univ-nantes.fr

***Abstract*—Data sharing services on the web host huge amounts of resources supplied and accessed by millions of users around the world. While the classical approach is a central control over the data set, even if this data set is distributed, there is growing interesting in decentralized solutions, because of good properties (in particularity, privacy and scaling up). In this paper, we explore a machine learning side of this work direction. We propose a novel technique for decentralized estimation of probabilistic mixture models, which are among the most versatile generative models for understanding data sets. More precisely, we demonstrate how to estimate a global mixture model from a set of local models. Our approach accommodates dynamic topology and data sources and is statistically robust, i.e. resilient to the presence of unreliable local models. Such outlier models may arise from local data which are outliers, compared to the global trend, or poor mixture estimation. We report experiments on synthetic data and real geo-location data from Flickr.**

***Keywords*-Probabilistic mixture models ; Distributed data ; Decentralized estimation ; Gossip ; Robust estimation.**

I. Introduction

Web-based content sharing services gather huge amounts of data (multimedia documents and associated meta-data tags, scientific data, etc...) uploaded and retrieved by many users around the world. The present work is located at the crossroad of the two following research lines in this field:

- the large amount of user-contributed data is amenable to statistical analysis. For instance, a classical goal is content recommendation through collaborative filtering [1]. Another example pertains to collaborative tagging of images [2], which has been identified as a precious resource for statistical learning models of visual appearance, where the lack of labelled data was long a bottleneck [3];
- many popular systems rely on a central control on the data set, even if the data is physical distributed for performance and data availability reasons. Yet, there are strong grounds for promoting alternative solutions, founded on a distribution of data storage with a decentralization of control over information flow. Compared to centralized systems, they demonstrate good properties in terms of scaling up to large amounts of data and avoid having a single point of failure. Further, they can offer participant's control over propagation and exploitation of their personal data [4], and lead to novel approaches to information retrieval [5].

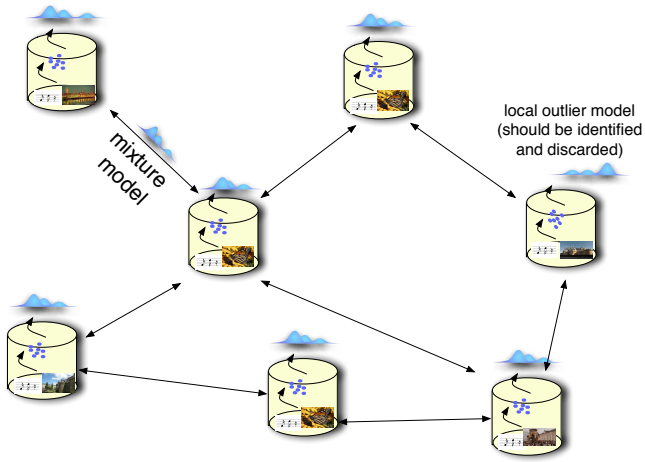


Figure 1. Overview of the proposed scheme: Mixture models are first estimated locally on local data. They are then exchanged and aggregated with neighbouring models, though a gossip process. Simultaneously, outlier models identified are progressively discarded.

While we are far from addressing all of these issues here, our point is that statistical machine learning in decentralized data management systems is becoming a key research track. In this perspective, this paper focuses on learning a highly popular probabilistic model for continuous data, whether for density modelling or data clustering, the Gaussian mixture model [6]. Discrete counterparts, such pLSA [7] and extensions, are also widely used for topic modelling and community identification. Mixture models are of the semi-parametric type, i.e. they have a built-in trade-off between the ability to model a wide range of data distributions and the parsimony of representation.

Let us assume a set of nodes, each node hosting a local data set. As sketched on fig. 1, let us assume that, indepen-

dently at each node, a standard mixture model estimation technique is applied on local data, supplying a *local model*. The goal we address is the estimation of a *global mixture model*, reflecting the probability density for the global data set. Because we carry out this task using mixture models, we in fact perform clustering of the data at the same time.

In an original fashion, we operate through aggregation of local mixture models and propagation of these by means of a gossip protocol. Gossip (or epidemic) protocols are a means of disseminating information in a decentralized fashion, which are well suited to cope with volatile peers and evolving topology [8]. As node join and leave the system, our propagation/aggregation mechanism updates the global estimate accordingly. Let us underline that model aggregation may be achieved by accessing only mixture model parameters (i.e. statistics), without needing to read local data nor transfer it over the network.

The main contribution of this paper is a technical solution for estimating a global model by dynamically combining local estimates, while being resilient to the presence of unreliable local models. In other words, we conduct statistically robust global model distributed estimation, from local models. Such outlier models may arise from local data which are outliers, compared to the global trend, or poor mixture estimation. In any case, such local models should be discarded from the dynamic estimation process.

Robust estimation is a classical chicken-and-egg problem: correct global model computation requires outliers to be identified and discarded, but their identification requires some reasonable estimate of the global model they deviate from. Typically, M-estimators solve this iteratively. With regard to this baseline situation, the original of our problem is two fold: we should estimate a model from a set of probability distributions ; the scheme for outlier identification and rejection is decentralized.

The remainder of the paper is organized as follows. Section 2 presents related work. Section 3 disclosed the technical details of our solution. Section 4 reports experimental result, while section 5 concludes and provides perspectives.

II. Related Work

Numerous decentralized clustering approaches have been proposed in the literature [9]–[13]. Roughly, the mechanism consists in iterating between two steps: first an aggregation function is applied independently on each node, and second, a communication protocol is applied to communicate the local clustering parameters on the network. The process stops when a convergence criterion is achieved, based for example, on the parameter similarity on each node. The differences between the various proposals concern the underlying distributed environment (P2P network, sensor network,...), the message exchanged between the nodes (representatives data, models parameters,...) and the number of rounds of communication required by the algorithm.

In the context of the estimation of a model mixture, several works [10], [11] propose a distributed EM algorithm. In both cases, a node applies local iterations of the Expectation and Maximisation steps and then sends its model parameters to its neighbors until convergence. The main difference with our approach is the access to the data set. Indeed, in our context we make the hypothesis that the local Gaussian models are already available, so that our estimation process does not rely directly on the distributed data sets.

Several works [14] also propose decentralized algorithms robust to outliers, an important property for our objective. The principle is similar to the precedent works, but integrates a step to detect the outliers. To our knowledge, the methods proposed in this context are based on existing approaches already used in the scope of centralized clustering. Thus, the nearest neighbor [14] or statistical [14], [15] approaches may well be applied. An interesting method [15], presenting the advantage to avoid critical arbitrary parameters, is based on a sampling approach: several clusterings are carried out on different data samples and the best clustering obtained is the closest one of the whole data set. Our proposal detection method of outliers is based on this approach but applied directly on probabilistic models.

Finally, after applying an aggregation function, a communication protocol must be applied to obtain the global model. Indeed, in our solution, each node iterates between two steps: it collects all the models of its neighbors list in order to re-compute its local model. Then, it selects one node from its neighbors to update its list. The choice of the selected node to communicate with and the way to swap the list of neighbors is crucial to enable a correct dissemination of the information. For that, we use the gossip protocol which is a probabilistic way to choose a member pairs to communicate and exchange an information [16]. Here, we use a specific implementation: the peer sampling service (PSS) [17]. The advantage of such a solution is practically to improve the aggregation convergence [16]. It proposes to update continuously the overlay topology of the network, in order to make it dynamic and more realistic than the static topologies. In a peer to peer network, nodes can leave or join the system continuously.

III. A distributed and robust estimation algorithm

Our proposal is founded on a decentralized and robust algorithm running independently on each node of the network. Let us assume a network composed of n nodes, where each node i is defined with the following properties:

1) a local data set D_i;
2) a mixture model M_i;
3) a list of c neighbors L_i.

The models M_i are a precondition for our algorithm: we assume data set D_i on each node is modelled as a probabilistic

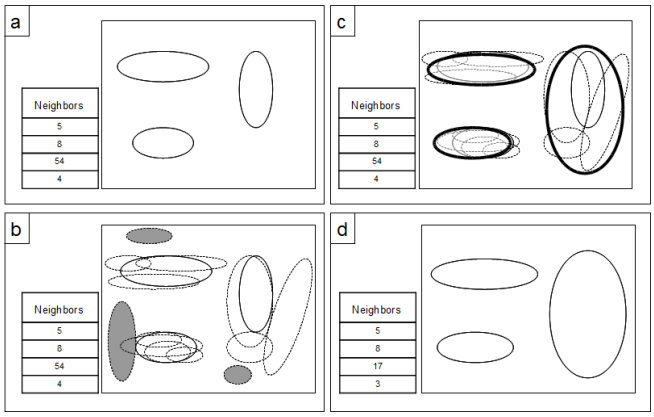


Figure 2. Example of our decentralized and robust algorithm: figure a shows an initial node i with its Gaussian model in two dimensions and its list of neighbors L_i. The first step consists in retrieving the Gaussian models of L_i, represented by the dashed ellipses on figure b. A detection of the outlier models is then carried out on this set of models: the result of the detection is showed by the model composed of 3 components depicted by the plain gray ellipses. This outlier is not used to compute the new estimation of the current node. Figure c presents the new aggregation model obtained (dark ellipse). Finally, the last step is to change the topology of the network by updating a part of the initial neighbors. Neighbors 4 and 54 are replaced by neighbors 3 and 17 on figure d.

mixture, using for instance an EM algorithm or a Bayesian form thereof, supplying model parameters M_i.

Overall, the scheme iterates over the 4 following steps. For node i:

- **retrieve all the models of its neighbors** L_i**.** Each node $j \in L_i$ sends its local model M_j to node i. Let $M_{neighbor}$ be the set of L_i's models $\cup$ model M_i;
- **detect and outliers locally**: filtering outliers out of $M_{neighbor}$ is carried out with a model clustering process. The chosen algorithm is adapted from a sampling method [15]. Note that node i itself can be considered as an outlier. Let M_{robust} be the set of models built from $M_{neighbor}$ where outlier models have been removed;
- **update its model** M_i: we aggregate the set M_{robust} to obtain a new local model. To this aim, we resort to a technique exploiting approximate KL divergences between mixtures [18], that enables to aggregate mixture models. It operates similarly to a k-means algorithm: given an initial set of mixture and a number of components P, it provides a final mixture of P components;
- **apply a gossip protocol** : in order to guarantee a correct propagation of the new model M_i, the node i updates its list of neighbors. The proposed method is founded on the peer sampling service [17]. In a nutshell, the principle consists in exchanging a part of the neighbors list of M_i with one of its neighbors. The point is to increase the convergence speed of the clustering process and the quality of the global model estimate by changing the topology of the network. Each new aggregation of models for model M_i is then carried out with different neighbors, avoiding to fall in a local configuration;
- **compute the convergence criterion**: before executing a new iteration of the estimation process, we check whether models in the neighborhood of M_i are all similar enough.

The main algorithm and an example of the different steps are presented respectively in algorithm 1 and on figure 2.

In the following, we explain in details the different steps of our algorithm. The similarity measure between mixture models is first presented, followed by the detection of outliers and the estimation algorithm of Gaussian models. Finally, the gossip protocol used to change the topology of the network and the convergence criterion are explained.

A. Similarity between Gaussian components

In order to aggregate or filter out outliers among a set of Gaussian models, we resort to an approximation of the Kullback-Leibler divergence. Our solution was motivated by the work proposed in [19], which assessed several approaches to compare Gaussian mixture models. As expected, it stated that Monte Carlo sampling leads to best accuracy, but at the price of a high calculation complexity. Thus, we focus on methods aiming the best trade-off between accuracy in KL approximation and computational cost (i.e. requiring only models parameters). Experiments from [19] conclude that the best approaches are the matched bound and the variational approximations. Because of its lower cost, we resort to the matched bound criterion.

$$KL_{match}(f||g) = \sum_i \pi_{f_i} \left(KL(f_i||g_{m(i)}) + log \frac{\pi_{f_i}}{\pi_{g_{m(i)}}} \right). \quad (1)$$

where π_{f_i} is the prior probability of a component f_i, and m is the matching function between elements of f and g

Algorithm 1 Iterative process applied on node i

Require: A mixture M_i, a list of neighbors L_i and the number of components P for the global model
while Convergence not achieved **do**
- Build the set of mixture $M_{neighbor}$ composed of M_i and the set of models contained in its neighbors L_i
- Filter the outlier out from $M_{neighbor}$: M_{robust} = REMOVE_OUTLIER($M_{neighbor}$)
- Aggregation of M_{robust}: M'_i = ESTIMATION(M_{robust_i}, P)
- Update of the neighbors L_i with the peer sampling service
- Compute the convergence criterion

end while

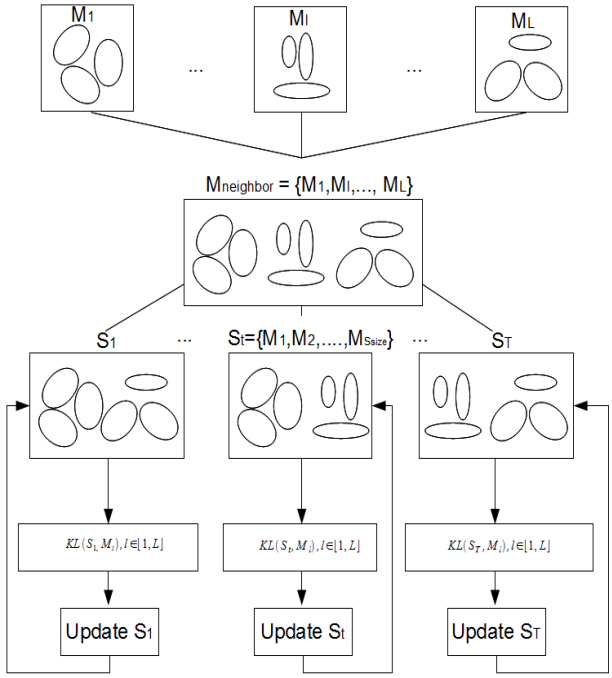


Figure 3. Example of the sampling algorithm: first, a node i retrieves the models of its neighbors and T samples are drawn from the set $M_{neighbor} = \{M_1, M_l, \ldots, M_L\}$ obtained. Second, for each sample an iterative algorithm is applied to improve their quality: the goal is to except in each sample the outlier models. For a sample S_t, the iterative algorithm consists in comparing the similarity between the components $M_l \in M_{neighbor}$ and those in S_t. The nearest components $M_l \in M_{neighbor}$ are then selected and used to update the sample S_t. The convergence is achieved when the samples keep stable.

defined as follows:

$$m(i) = \arg\min_j (KL(f_i||g_j) - log\pi_{g_j}) \quad (2)$$

where π_{g_j} is the prior probability of a component g_j and KL is defined as:

$$KL(f_i||g_j) = \frac{1}{2}[log\frac{|\Sigma_{g_j}|}{|\Sigma_{f_i}|} + Tr|\Sigma_{g_j}^{-1}\Sigma_{f_i}| - \gamma + (\mu_{f_i} - \mu_{g_j})^t\Sigma_{g_j}^{-1}(\mu_{f_i} - \mu_{g_j})] \quad (3)$$

where γ is the dimension of the feature space.

This convenient criterion is used as a similarity distance in both our sampling algorithm and our aggregation algorithm.

B. Detection of outliers

Our algorithm is an adaptation of the sample algorithm [15] to deal with Gaussian models. Its objective is to build a sample with no outlier in order to obtain a baseline before to apply our aggregation algorithm. The principle is to draw randomly several samples of a set of Gaussian model, to improve them with an iterative algorithm and to select the one presenting the best similarity with the initial data set. Such a method presents interesting advantages for our purpose:

- it does not depend on strong arbitrary parameters: it is based on a simple hypothesis, the percentage of outliers supposed on the network;
- it does not depend on the space dimension or domain of the initial data set: our choice is then more convenient than for example distance-based algorithms using a fix threshold to detect outliers.

In our solution, we use the KL_{match} (eq. 1) as a similarity criterion. Figure 3 presents an overview on this algorithm.

More formally, the initial step of the algorithm is to draw T samples of size s_{size} from a set of Gaussian model $M_{neighbor} = \{M_1, \ldots, M_L\}$. Then for each sample $S_t = \{M_1, \ldots, M_r, \ldots, M_{s_{size}}\}, t \in [1, T], M_r \in M_{neighbor}$, the following steps are applied until convergence:

1) compute a model M_c obtained with the merge of all $M_r \in S_t$;
2) compute the KL_{match} divergence between M_c and each model $M_l \in M_{neighbor}$. A ascending sorted list of models is then computed;
3) update the sample S_t with the first s_{size} of the sorted list obtained previously.

The convergence is achieved when each sample S_t remains stable. Finally, the sample minimizing its KL_{match} divergence with all the M_l is selected. This model is supposed to be cleared of outliers and is then used as a baseline for the final aggregation of the set of models $M_{neighbor}$.

This algorithm depends on two understandable parameters, the size s_{size} of each sample and their number T:

- the setting of the parameter s_{size} has to reflect the average number of outlier on the network, but of course this information is unknown. Practically, the higher the value of s_{size}, the higher is the chance that the sample contains outliers. And inversely, the lower its value, the higher is the risk to select a set of models none representative of the real global view. The setting of this parameter is then a compromise between the robustness and the quality of the final sample. But note that a wrong initialization of s_{size} can be balanced with a higher number of generated samples T, so that an uncertainty is tolerated;
- the impact of the parameter T concerns the performance: a high value would test a large number of samples but it will increase the computational cost. To improve this point, algorithm [15] proposes to make a first selection of the best samples. The previous iterative algorithm is first applied twice on a large number of samples, leading to a selection of a subset of the best ones. The algorithm is then applied until convergence only on the selected samples.

Algorithm 2 REMOVE_OUTLIER: Sampling algorithm to remove outlier in a set of mixture models

Require: A set of Gaussian mixture $M_{neighbor} = M_1, ..., M_L$ and the number of components P for the global model

- Draw randomly T samples $S_{Sample} = S_1, ..., S_T$ composed of s_{size} Models of $M_{neighbor}$

for each sample S_t, $1 < t < T$, repeat the following operations twice **do**

  for each model $M_j \in M_{neighbor}$, $1 < j < L$ **do**

    - Compute the KL_{match} divergence between M_j and M_t the concatenated model of all $M_r \in S_t$, $M_t = \frac{1}{\pi}\sum_{r=1}^{s_{size}} M_r$, where $M_r = \sum_{i=1}^{m_i} \pi_r^i N^i$, $\pi = \sum_{r=1}^{s_{size}} \pi_r^i$, and N^i is a Gaussian component

  end for

  - $S_t = \{ M_r \mid r = \arg\min_{1<j<L} (KL_{match}(M_t, M_j))\}$ and $1 < |S_t| < s_{size}$

end for

- Select from S_{Sample} the T' best samples S_{best} minimizing $\sum_{j=1}^{L} KL_{match}(M_t, M_j)$

for each sample $S_t \in S_{best}$, $1 < t < T'$ **do**

  repeat

    for each model M_j, $1 < j < L$ **do**

      - Compute $KL_{match}(M_t, M_j)$

    end for

    - $S_t = \{ M_r \mid r = \arg\min_{1<j<L} (KL_{match}(M_t, M_j))\}$ and $1 < |S_t| < s_{size}$

  until S_t is unchanged

end for

return the sample S_t, $1 < t < T'$, minimizing $\sum_{j=1}^{L} KL_{match}(M_t, M_j)$

Algorithm 2 details the sampling algorithm used to remove outliers from our aggregation process.

C. Mixture Model Estimation

Let the problem be formulated as transforming a mixture model f into another mixture g with less components, while minimizing a Kullback-Leibler divergence involved by the simplification process. This section recalls how this was solved in [18], as the sole difference in our approach is the use of a different KL-approximation, the KL_{match}. A key feature of their solution is that only model parameters are accessed to group components, i.e. neither access to data nor sampling are required. Thus, it is very cost effective in terms of computation.

The search for optimal g is composed in two alternating steps. The first one consists in determining the best mapping m between the components of f and g such that criterion (4) is minimized :

$$\begin{aligned} d(f,g) &= d(f,g,m) \\ &= \sum_i KL_{match}(f_i || g_{m(i)}) \end{aligned} \tag{4}$$

where $m = \arg\min_{m'} \quad d(f,g,m')$.

The second step updates the model parameters of g, again from the sole model parameters of f. It is proposed to compute the average of the mean and the covariance in accordance with the mapping m obtained at previous step. Each Gaussian is updated as follows:

$$\mu_{g_j} = \frac{1}{\pi_{g_j}} \sum_{i \in m^{-1}(j)} \pi_{f_i} \mu_{f_i} \tag{5}$$

$$\Sigma_{g_j} = \frac{1}{\pi_{g_j}} \sum_{i \in m^{-1}(j)} \pi_{f_i} \left((\Sigma_{f_i} + (\mu_{f_i} - \mu_{g_j})(\mu_{f_i} - \mu_{g_j})^T) \right) \tag{6}$$

where $\pi_{g_j} = \sum_{i \in m^{-1}(j)} \pi_{f_i}$.

These two steps are iterated until the convergence of the criterion defined in equation 4.

As our aggregation process is similar to a k-means algorithm, let us discuss major shortcomings:

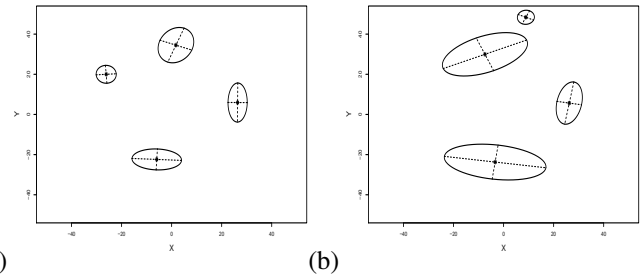
- the number of components P of the reduced model must be known. This is the sole critical parameter of our algorithm. Comparing several complexities of the reduced model is nevertheless possible with Bayesian criteria which in our context, should be adapted to work directly on mixture parameters, rather than data. This is discussed in another paper [**?**].
- the result is highly dependent on the parameter initializations: we opt for the kmeans++ [20] to set the initial parameter of the reduction model P. This method consists in selecting the centers as far as possible from each others, while avoiding outliers. Adapted to the context of the mixture model, this method consists in first, selecting a center randomly from the initial components, and second, choosing the subsequent cluster center from the remaining components with a probability proportional to its distance squared to the closest center. The distance used between components is again the KL divergence.

D. Update of the network topology

In order to increase the convergence of our clustering process, we propose to use an implementation of a gossip protocol, the peer sampling service [17]. This protocol consists in changing dynamically the topology of the network to improve the propagation of the information. In our context, the goal is to improve the propagation of the aggregated models obtained iteratively on each node.

Let's recall that each node has a list of its neighbors composed of c elements. The different steps of the protocol for a node i are the following:

1) select randomly a node j from the neighbors list of the node i to initiate a communication;



(a) (b)

Figure 4. figure (a) shows the global view obtained without the outliers models while figure(b) presents the global view obtained with the outliers models.

2) exchange half of the neighbors list of the node i with the node j. Both the list are then updated with a new set of nodes;
3) discard the duplicate and reduce the size back to c (the same step is applied on j).

This protocol enables to increase the convergence of our global view as well as its quality. Indeed, the exchange of neighbors enables to avoid local concentration of outlier models that would lead to bias the aggregation process. This dynamic property presents then the advantage to dispatch randomly the outliers in all the network, leading to increase the chance to detect them with our sampling algorithm.

E. Convergence criterion

Our objective is to stop automatically our algorithm when each node contains the same global estimation. We then propose a decentralized solution to detect such a configuration. Roughly, our solution consists in comparing for each node its local models with the ones of its neighborhood: if they all present a strong similarity, our estimation process on this node is interrupted.

More precisely, let i be a node containing its new update model M_i the method consists in:

1) sorting its neighbors in ascending order in accordance with the KL divergence between M_i and their models;
2) selecting the model $M_{threshold}$ defined as the S_{size}th nearest neighbor in the list obtained. Recall that S_{size} is the average number of outlier in the network. We then suppose that the selected models is the last good models among its neighbors;
3) comparing KL_{match} of $M_{threshold}$ to a threshold ϵ: if KL_{match} is lower the algorithm is stopped, else a new iteration is processed.

The parameter ϵ is set empirically to a small value: a $KL_{match}(M_i, M_j)$ divergence lower to ϵ must involve a strong similarity between M_i and M_j.

IV. Experiments

To evaluate our proposed algorithm, we report experiments on, first a synthetic data set with known ground truth and then on a real geolocation data set taken from Flickr. These experiments aim at computing the global model from the distributed models. The objective is to evaluate different aspects of our solution as the robustness and the aggregation process. For both experiments, we ran our algorithm on a peer to peer network composed of 200 nodes, characterized as follows:

- each node has $c = 20$ neighbors;
- each node holds a Gaussian mixture model;
- 20% of models are considered as outliers;
- the number of nodes exchanged with the gossip protocol is set to $c/2$, the half of the neighbors list.

A. Experimental results on synthetic data

The objective of this first experiment is to assess the robustness of our distributed algorithm. Our synthetic data set is generated as follows:

- each node contains a Gaussian mixture model with a number of components varying between 3 and 12;
- the components of the outlier nodes are all generated randomly.

In order to evaluate the results of our distributed algorithm, we compute our global model without the outliers in a centralized way, presented in figure 4(a). This model, composed of 4 distinct components, represents the true global data distribution over the network. To demonstrate that our sampling algorithm succeed to discard the outliers, we also compute the global view with the outliers models, depicted on figure 4(b). The result shows the impact of the outliers on the aggregation process: the models obtained present strong differences.

Figure 5 presents an example of initial models of 4 nodes on the peer to peer network. These models are composed of different number of components (Figure 5(a),(b),(c) and (d) with 7, 9, 2 and 6 components respectively). An example of outlier models is showed in figure 5(c).

Figure 7 presents the evolution of our convergence criterion all along the iterations of our algorithm on each node. The convergence criterion used here is the average of the KL divergence between each local model and the real global model. This curve decreases after each iteration and tends quickly toward zero. This convergence is confirmed by the results obtained on figure 6 of the previous models (figure 5) after 13 iterations of our algorithm. All the models, even the outliers (figure 6(c)), seem identical to the initial centralized global model without outliers (figure 4(a)). This result confirms that our sampling algorithm is effective to discard the outliers from the aggregation process.

With this experiment, we assessed the robustness and convergence properties of our distributed algorithm to obtain a real global view of the network.

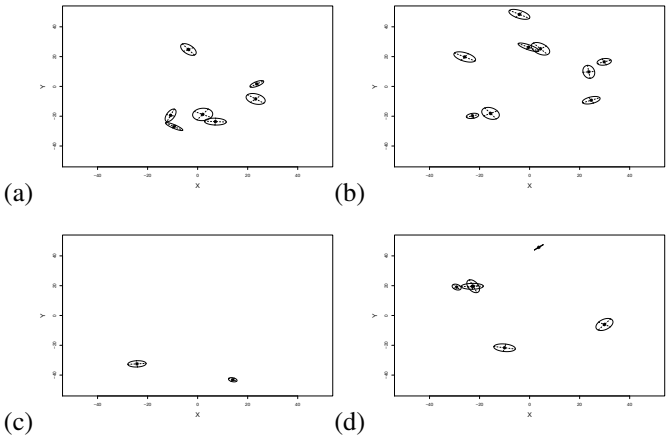


Figure 5. These figures present the initial model of 4 nodes of the peer to peer network

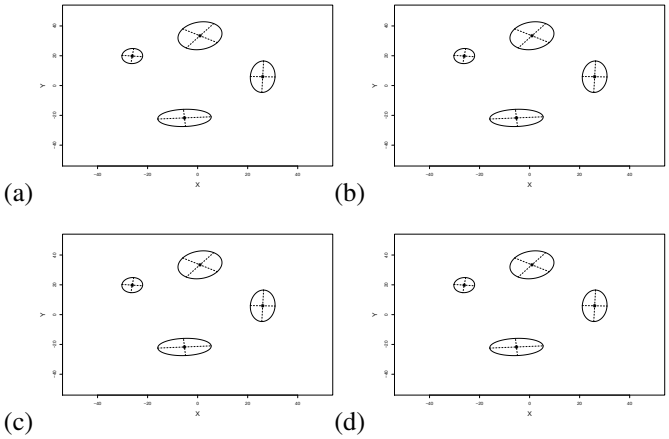


Figure 6. These figures present the models of the 4 nodes presented in figure 5 after 13 iterations

B. Experimental results on real data

This second experiment proposes to assess our algorithm on a real multimedia set: a collections of $406,450$ geotagged Flickr images from $5,951$ users, presented in figure 8. To apply our algorithm, we first split this collection on 200 nodes, each one containing a set of data varying from 500 to 4000 images. The local Gaussian models are obtained with the EM algorithm and their complexity are determined thanks to the BIC criterion (models were tested between 3 and 6 components).

Figure 10 presents an example of initial models of 2 nodes obtained after applying the EM algorithm on their local data set. The two models presented here are quite different, due to their different initial data set: node of figure 10(a) is composed of 3 components while the one of figure 10(b) contains 6 components. Their models obtained after 10 iterations of our algorithm are presented in figures 11. Even though these nodes present initial differences, they converge

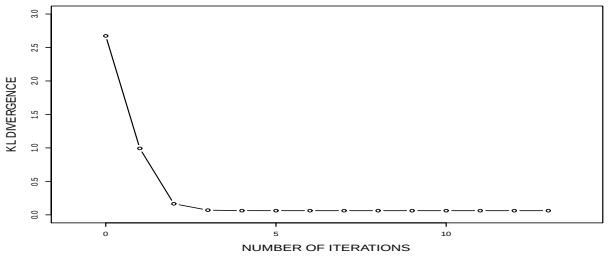


Figure 7. This figure shows the average KL divergence between the model on each node and the initial global view.

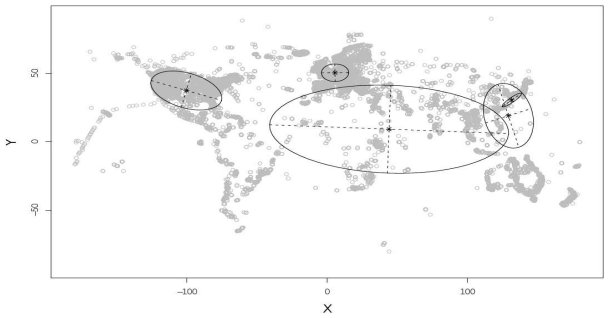


Figure 8. This figure presents the localisation of all the images and the global density estimation obtained in a centralized way.

to a similar model with an identical number of components. Figure 9 shows that, similarly to the previous experiment, the KL criterion converges quickly toward zero.

In order to assess our decentralized algorithm, we compute the aggregation of all the local Gaussian models in a centralized manner, presented on figure 8. The comparison between figure 8 and figure 11 shows that both the models obtained are similar. We can then conclude that our algorithm succeed to build a correct global estimation in a decentralized way.

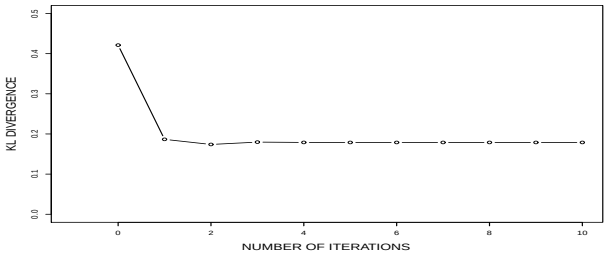
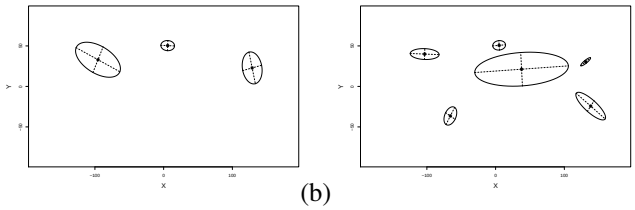
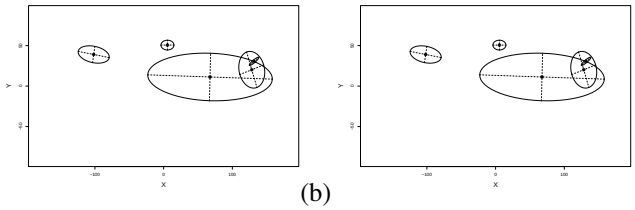


Figure 9. This figure shows the average KL divergence between the model on each node and the centralized global view.



(a) (b)

Figure 10. These figures present the initial models of 2 nodes after applying locally the EM algorithm on their data sets.



(a) (b)

Figure 11. These figures present the two models of figure 10 after 10 iterations of our estimation algorithm.

V. Conclusion

This paper presents an original solution for decentralized and statistically robust learning of a probabilistic mixture model, from a distributed data set. Such settings are of much current interest in decentralized data sharing systems, for data understanding, content recommendation and so forth. The proposed scheme is founded on local aggregation of models, that rest upon the sole use of model parameters, learning to interesting properties with regard to data privacy and network load, rather than original data. We then illustrated the proposal on synthetic and real-world results.

We are currently adapting the proposal to bi-clustering mixture models, such as latent Dirichlet allocation models, that are fundamental to topic-based clustering for a wide range of applications. Besides, while in practice we have observed good convergence properties, their existence and characteristics remain to be established theoretically. Finally, recent advances pertaining to gossip algorithms should be introduced in the scheme.

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 734–749, 2005.

[2] K.-H. Yap, K. Wu, and C. Zhu, "Knowledge propagation in collaborative tagging for image retrieval," *Signal Processing Systems*, vol. 59, no. 2, pp. 163–175, 2010.

[3] J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, *Toward Category-Level Object Recognition (Lecture Notes in Computer Science)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[4] A. Shikfa, M. Önen, and R. Molva, "Privacy and confidentiality in context-based and epidemic forwarding," *Computer Communications*, vol. 33, no. 13, pp. 1493–1504, 2010.

[5] A. Boutet, D. Frey, R. Guerraoui, and A.-M. Kermarrec, "Whatsup: News, from, for, through, everyone," in *Peer-to-Peer Computing*. IEEE, 2010, pp. 1–2.

[6] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[7] T. Hofmann, "Probabilistic latent semantic analysis," in *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm, 1999.

[8] A.-M. Kermarrec and M. van Steen, "Gossiping in distributed systems," *Operating Systems Review*, vol. 41, no. 5, pp. 2–7, 2007.

[9] S. Datta, C. Giannella, and H. Kargupta, "K-Means Clustering over a Large, Dynamic Network," in *Proceedings of 2006 SIAM Conference on Data Mining*, Bethesda, MD, April 2006.

[10] D. Gu, "Distributed EM algorithm for Gaussian Mixtures in Sensor Networks," *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1154–1166, 2008.

[11] W. Kowalczyk and N. A. Vlassis, "Newscast EM," in *NIPS*. MIT Press, 2004, pp. 713–720.

[12] B. Safarinejadian, M. B. Menhaj, and M. Karrari, "A distributed EM algorithm to estimate the parameters of a finite mixture of components," *Knowl. Inf. Syst.*, vol. 23, no. 3, pp. 267–292, 2010.

[13] J. Wolfe, A. Haghighi, and D. Klein, "Fully distributed EM for very large datasets," in *ICML*, 2008, pp. 1184–1191.

[14] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009.

[15] P. J. Rousseeuw and K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999.

[16] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *ACM Trans. Comput. Syst.*, vol. 23, no. 3, pp. 219–252, 2005.

[17] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen, "Gossip-based peer sampling," *ACM Trans. Comput. Syst.*, vol. 25, no. 3, p. 8, 2007.

[18] J. Goldberger and S. T. Roweis, "Hierarchical clustering of a mixture model," in *NIPS*. MIT Press, 2004, pp. 505–512.

[19] J. R. Hershey and P. A. Olsen, "Approximating the Kullback Leibler divergence between Gaussian mixture models," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, 2007, pp. IV–317–IV–320.

[20] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.